Flexible Load Balancing with Multi-dimensional State-space Collapse: Throughput and Heavy-traffic Delay Optimality

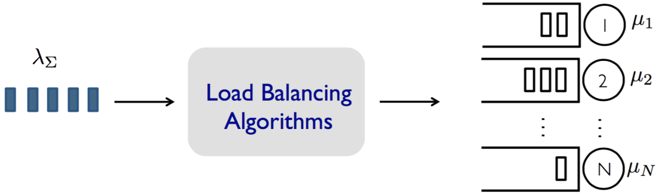
Xingyu Zhou

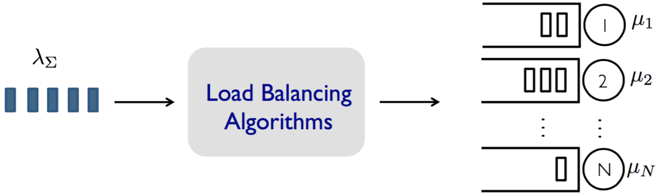
The Ohio State University

Joint work with...

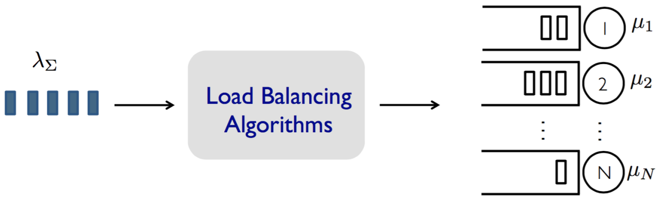
Jian Tan, OSU

Ness Shroff, OSU


λ_Σ
Load Balancing Algorithms
1
μ_1
2
μ_2
N
μ_N

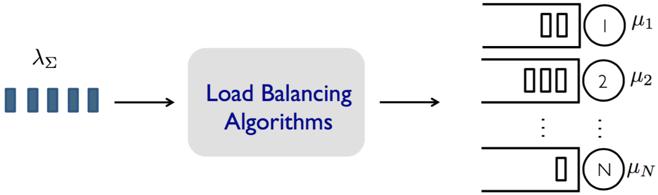


- Discrete-time system, i.e, time-slotted.



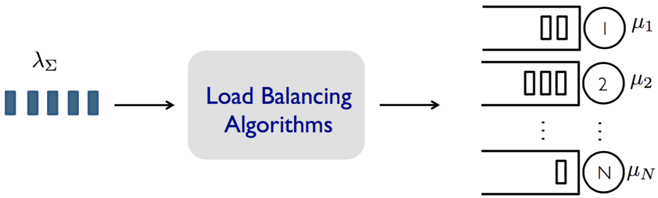
- Discrete-time system, i.e, time-slotted.
- Arrival rate at each time slot is λ_Σ, arbitrary distribution [1] .

[1] with exponential decay tail



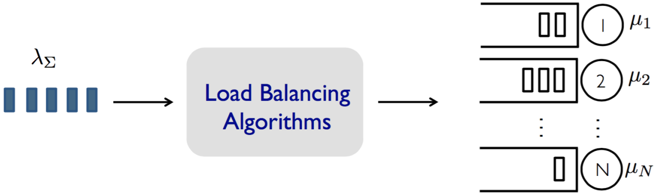
- Discrete-time system, i.e, time-slotted.
- Arrival rate at each time slot is λ_Σ, arbitrary distribution [1] .
- Service rate at each server k is μ_k, arbitrary distribution.

[1] with exponential decay tail



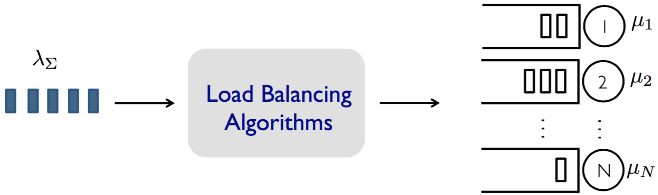
- Discrete-time system, i.e, time-slotted.
- Arrival rate at each time slot is λ_Σ, arbitrary distribution [1] .
- Service rate at each server k is μ_k, arbitrary distribution.
- Arrival and service are independent.

[1] with exponential decay tail



The goal of load balancing:

choose the *right* server(s) for each request.



The goal of load balancing:

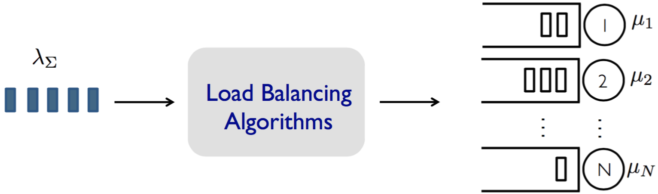
choose the *right* server(s) for each request.

What does *right* mean?

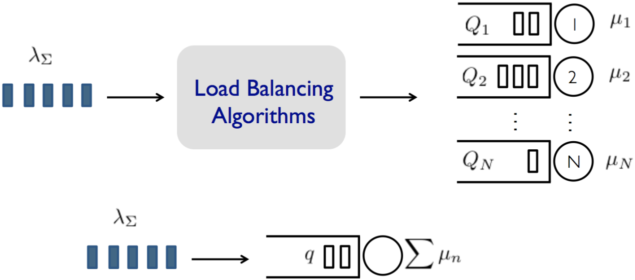
Throughput Optimality

Definition

It can stabilize the system for any arrival rate in capacity region, i.e, for any $\epsilon > 0$ where $\epsilon = \sum \mu_n - \lambda_\Sigma$.



Heavy-traffic Delay Optimality

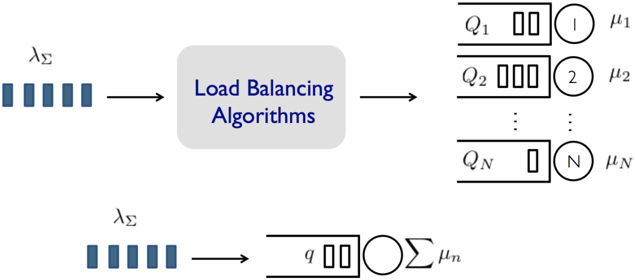


Fact: $\mathbb{E}\left[\sum Q_n\right] \geq \mathbb{E}\left[q\right]$, since packet remains in the queue until finished.

Heavy-traffic Delay Optimality

Definition

It can achieve the lower bound on delay when $\epsilon \to 0$, that is,
$\lim_{\epsilon \downarrow 0} \epsilon \mathbb{E}\left[\sum Q_n\right] = \lim_{\epsilon \downarrow 0} \epsilon \mathbb{E}\left[q\right]$ (since the queue length is order $O(1/\epsilon)$)



Fact: $\mathbb{E}\left[\sum Q_n\right] \geq \mathbb{E}\left[q\right]$, since packet remains in the queue until finished.

Some 'optimal' policies...

Some 'optimal' policies...

- **Join-Shortest-Queue (JSQ)**: Sample all the queue lengths, join the shortest one. [Foschini and Salz'78], [Eryilmaz and Srikant'12]

Some 'optimal' policies...

- **Join-Shortest-Queue (JSQ)**: Sample all the queue lengths, join the shortest one. [Foschini and Salz'78], [Eryilmaz and Srikant'12]
- **Power-of-d choices (Pod)**: Randomly sample d queues, join the shortest one. [Chen and Ye'12], [Maguluri, et al'14]

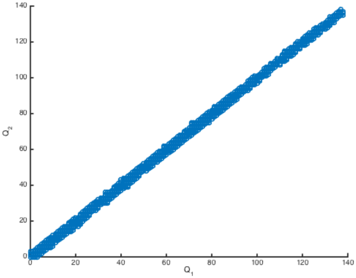
Some 'optimal' policies...

- **Join-Shortest-Queue (JSQ)**: Sample all the queue lengths, join the shortest one. [Foschini and Salz'78], [Eryilmaz and Srikant'12]
- **Power-of-d choices (Pod)**: Randomly sample d queues, join the shortest one. [Chen and Ye'12], [Maguluri, et al'14]
- **A general class of optimal policies:** Any policy that statistically prefers shorter queues is heavy-traffic optimal. [Zhou, et al'18]

Some 'optimal' policies...

- **Join-Shortest-Queue (JSQ)**: Sample all the queue lengths, join the shortest one. [Foschini and Salz'78], [Eryilmaz and Srikant'12]
- **Power-of-d choices (Pod)**: Randomly sample d queues, join the shortest one. [Chen and Ye'12], [Maguluri, et al'14]
- **A general class of optimal policies:** Any policy that statistically prefers shorter queues is heavy-traffic optimal. [Zhou, et al'18]

All of them share one thing in common: state-space collapse to the line.

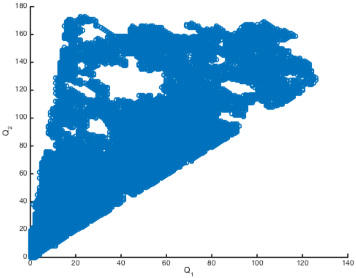


All the queue lengths are nearly equal in heavy traffic.

Warm-up...

Is it possible to achieve delay optimality in heavy traffic with the following state-space collapse?

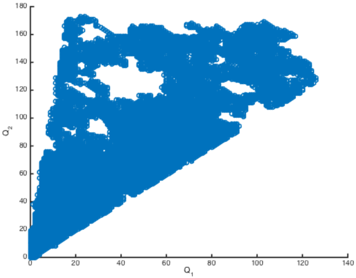
(A). Yes (B). No



Warm-up...

Is it possible to achieve delay optimality in heavy traffic with the following state-space collapse?

(A). Yes (B). No



The answer is Yes!

Part I: From single to multi-dimension state-space collapse.

Multi-dimensional cone...

- Consider the following finitely generated cone:

$$\mathcal{K}_\alpha = \left\{ \mathbf{x} \in \mathbb{R}^N : \mathbf{x} = \sum_{n \in \mathcal{N}} w_n \mathbf{b}^{(n)}, w_n \geq 0 \text{ for all } n \in \mathcal{N} \right\}, \quad (1)$$

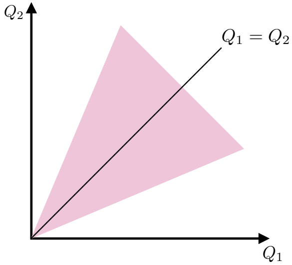
where $\mathbf{b}^{(n)}$ is an N-dimensional vector with the nth component being 1 and α everywhere, $\alpha \in [0, 1]$.

Multi-dimensional cone...

- Consider the following finitely generated cone:

$$\mathcal{K}_\alpha = \left\{ \mathbf{x} \in \mathbb{R}^N : \mathbf{x} = \sum_{n \in \mathcal{N}} w_n \mathbf{b}^{(n)}, w_n \geq 0 \text{ for all } n \in \mathcal{N} \right\}, \quad (1)$$

  where $\mathbf{b}^{(n)}$ is an N-dimensional vector with the nth component being 1 and α everywhere, $\alpha \in [0, 1]$.
- Example: $\mathbf{b}^{(1)} = (1, 0.5)$ and $\mathbf{b}^{(2)} = (0.5, 1)$



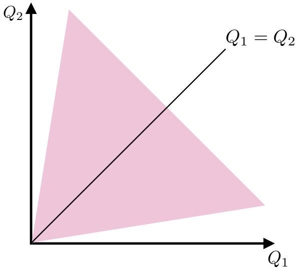
Multi-dimensional cone...

- Consider the following finitely generated cone:

$$\mathcal{K}_\alpha = \left\{ \mathbf{x} \in \mathbb{R}^N : \mathbf{x} = \sum_{n \in \mathcal{N}} w_n \mathbf{b}^{(n)}, w_n \geq 0 \text{ for all } n \in \mathcal{N} \right\}, \qquad (1)$$

  where $\mathbf{b}^{(n)}$ is an N-dimensional vector with the nth component being 1 and α everywhere, $\alpha \in [0, 1]$.

- Example: $\mathbf{b}^{(1)} = (1, 0.1)$ and $\mathbf{b}^{(2)} = (0.1, 1)$

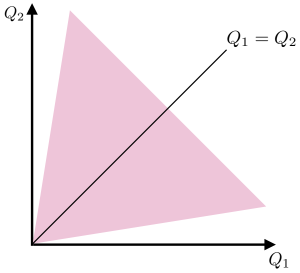


Multi-dimensional cone...

- Consider the following finitely generated cone:

$$\mathcal{K}_\alpha = \left\{ \mathbf{x} \in \mathbb{R}^N : \mathbf{x} = \sum_{n \in \mathcal{N}} w_n \mathbf{b}^{(n)}, w_n \geq 0 \text{ for all } n \in \mathcal{N} \right\}, \quad (1)$$

  where $\mathbf{b}^{(n)}$ is an N-dimensional vector with the nth component being 1 and α everywhere, $\alpha \in [0, 1]$.
- Example: $\mathbf{b}^{(1)} = (1, 0.1)$ and $\mathbf{b}^{(2)} = (0.1, 1)$



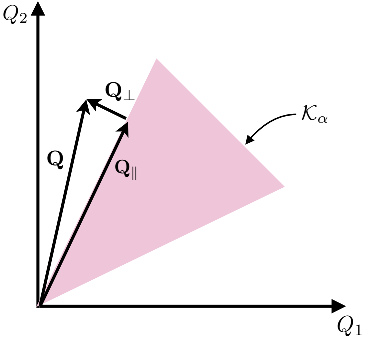
Smaller α, bigger cone.

State-space collapse to the cone...

- We can decompose the queue length vector as follows.

$$\mathbf{Q} = \mathbf{Q}_{\parallel} + \mathbf{Q}_{\perp},$$

as shown in



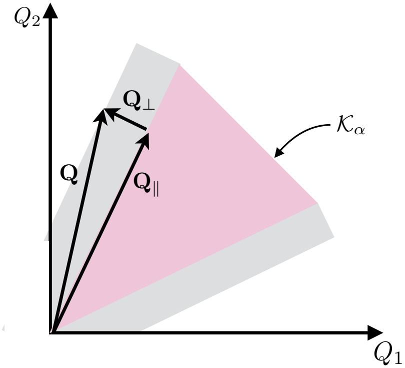
State-space collapse to the cone...

Definition

Let $\overline{\mathbf{Q}}$ be the steady-state, we say state-space collapses to $\mathcal{K}_\alpha$ if

$$\mathbb{E}\left[\left\|\overline{\mathbf{Q}}_{\perp}^{(\epsilon)}\right\|^r\right] \leq M_r \quad (2)$$

for all $\epsilon \in (0, \epsilon_0)$, $\epsilon_0 > 0$ and for each $r = 1, 2, \cdots$, M_r are constants that are **independent** of ϵ. (recall that ϵ is the heavy-traffic parameter.)



Main Result...

Theorem (Stability + Collapse to cone $\implies$ Optimality)

,

Main Result...

Theorem (Stability + Collapse to cone $\implies$ Optimality)

Given a throughput optimal load balancing policy,

,

Main Result...

Theorem (Stability + Collapse to cone $\implies$ Optimality)

Given a throughput optimal load balancing policy, if there exists an $\alpha \in (0, 1]$ *such that the state-space collapses to the cone* $\mathcal{K}_\alpha$,

Main Result...

Theorem (Stability + Collapse to cone $\implies$ Optimality)

Given a throughput optimal load balancing policy, if there exists an $\alpha \in (0,1]$ *such that the state-space collapses to the cone* $\mathcal{K}_\alpha$, *then this policy is heavy-traffic delay optimal in steady-state.*

Main Result...

Theorem (Stability + Collapse to cone $\implies$ Optimality)

Key implications:

- If $\alpha = 1$, the cone $\mathcal{K}_\alpha$ reduces to previous single dimensional line.

Main Result...

Theorem (Stability + Collapse to cone $\implies$ Optimality)

Key implications:

- If $\alpha = 1$, the cone $\mathcal{K}_\alpha$ reduces to previous single dimensional line.

Main Result...

Theorem (Stability + Collapse to cone $\implies$ Optimality)

Key implications:

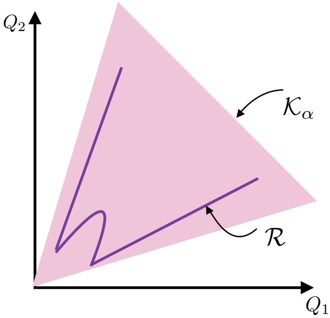
- If $\alpha = 1$, the cone $\mathcal{K}_\alpha$ reduces to previous single dimensional line.
- Delay optimality in heavy traffic does not require queue lengths being equal.

Main Result...

Theorem (Stability + Collapse to cone $\implies$ Optimality)

Key implications:

- If $\alpha = 1$, the cone $\mathcal{K}_\alpha$ reduces to previous single dimensional line.
- Delay optimality in heavy traffic does not require queue lengths being equal.
- The actual state-space collapse region $\mathcal{R}$ could even be non-convex.



Umm...it seems a little counter-intuitive, any intuitions?

The 'King' equation...

The sufficient and necessary condition for HT-optimality:

$$\lim_{\epsilon \downarrow 0} \mathbb{E}\left[\left\| \overline{\mathbf{Q}}^{(\epsilon)}(t+1) \right\|_1 \left\| \overline{\mathbf{U}}^{(\epsilon)}(t) \right\|_1 \right] = 0.$$

where the unused service vector $\mathbf{U}(t) = \max\{\mathbf{S}(t) - \mathbf{Q}(t) - \mathbf{A}(t), \mathbf{0}\}$.

The 'King' equation...

The sufficient and necessary condition for HT-optimality:

$$\lim_{\epsilon \downarrow 0} \mathbb{E}\left[\left\| \overline{\mathbf{Q}}^{(\epsilon)}(t+1) \right\|_1 \left\| \overline{\mathbf{U}}^{(\epsilon)}(t) \right\|_1 \right] = 0.$$

where the unused service vector $\mathbf{U}(t) = \max\{\mathbf{S}(t) - \mathbf{Q}(t) - \mathbf{A}(t), \mathbf{0}\}$.

- Note that $Q_n(t+1)U_n(t) = 0$ for all n and t.

The 'King' equation...

The sufficient and necessary condition for HT-optimality:

$$\lim_{\epsilon \downarrow 0} \mathbb{E}\left[\left\| \overline{\mathbf{Q}}^{(\epsilon)}(t+1) \right\|_1 \left\| \overline{\mathbf{U}}^{(\epsilon)}(t) \right\|_1 \right] = 0.$$

where the unused service vector $\mathbf{U}(t) = \max\{\mathbf{S}(t) - \mathbf{Q}(t) - \mathbf{A}(t), \mathbf{0}\}$.

- Note that $Q_n(t+1)U_n(t) = 0$ for all n and t.

IMPLICATIONS: No server is idle while others with high loads.

The 'King' equation...

The sufficient and necessary condition for HT-optimality:

$$\lim_{\epsilon \downarrow 0} \mathbb{E}\left[\left\|\overline{\mathbf{Q}}^{(\epsilon)}(t+1)\right\|_1 \left\|\overline{\mathbf{U}}^{(\epsilon)}(t)\right\|_1\right] = 0.$$

where the unused service vector $\mathbf{U}(t) = \max\{\mathbf{S}(t) - \mathbf{Q}(t) - \mathbf{A}(t), \mathbf{0}\}$.

- Note that $Q_n(t+1)U_n(t) = 0$ for all n and t.

IMPLICATIONS: No server is idle while others with high loads.

"*Probability theory is nothing but common sense reduced to calculation.*"

— Pierre Laplace

The 'King' equation...

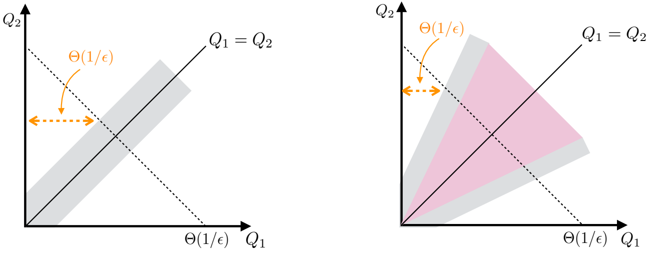
The sufficient and necessary condition for HT-optimality:

$$\lim_{\epsilon \downarrow 0} \mathbb{E}\left[\left\|\overline{\mathbf{Q}}^{(\epsilon)}(t+1)\right\|_1 \left\|\overline{\mathbf{U}}^{(\epsilon)}(t)\right\|_1\right] = 0.$$

where the unused service vector $\mathbf{U}(t) = \max\{\mathbf{S}(t) - \mathbf{Q}(t) - \mathbf{A}(t), \mathbf{0}\}$.

- Note that $Q_n(t+1)U_n(t) = 0$ for all n and t.

IMPLICATIONS: No server is idle while others with high loads.

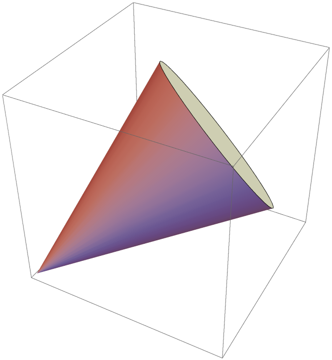


The problem with 'ice-cream' cone...

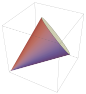
Consider the following cone given by

$$\mathcal{K}_\theta = \left\{ \mathbf{x} \in \mathbb{R}^N : \frac{\|\mathbf{x}_\parallel^{(\mathbf{1})}\|}{\|\mathbf{x}\|} \geq \cos(\theta) \right\},$$

where $\mathbf{x}_\parallel^{(\mathbf{1})}$ is the projection of $\mathbf{x}$ onto the line $\mathbf{1} = (1, 1, \ldots, 1)$.



The problem with ‘ice-cream’ cone...

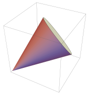


Consider the following cone given by

$$\mathcal{K}_\theta = \left\{ \mathbf{x} \in \mathbb{R}^N : \frac{\|\mathbf{x}_\parallel^{(\mathbf{1})}\|}{\|\mathbf{x}\|} \geq \cos(\theta) \right\},$$

where $\mathbf{x}_\parallel^{(\mathbf{1})}$ is the projection of $\mathbf{x}$ onto the line $\mathbf{1} = (1, 1, \ldots, 1)$.

The problem with 'ice-cream' cone...



Consider the following cone given by

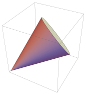
$$\mathcal{K}_\theta = \left\{ \mathbf{x} \in \mathbb{R}^N : \frac{\|\mathbf{x}_\parallel^{(\mathbf{1})}\|}{\|\mathbf{x}\|} \geq \cos(\theta) \right\},$$

where $\mathbf{x}_\parallel^{(\mathbf{1})}$ is the projection of $\mathbf{x}$ onto the line $\mathbf{1} = (1, 1, \ldots, 1)$.

Requirements: avoid one queue is empty while others are not.

- To exclude points on axes, e.g., (1,0,0), $\theta < \text{arccos}(1/\sqrt{3})$.

The problem with 'ice-cream' cone...



Consider the following cone given by

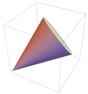
$$\mathcal{K}_\theta = \left\{ \mathbf{x} \in \mathbb{R}^N : \frac{\|\mathbf{x}_\parallel^{(\mathbf{1})}\|}{\|\mathbf{x}\|} \geq \cos(\theta) \right\},$$

where $\mathbf{x}_\parallel^{(\mathbf{1})}$ is the projection of $\mathbf{x}$ onto the line $\mathbf{1} = (1, 1, \ldots, 1)$.

Requirements: avoid one queue is empty while others are not.

- To exclude points on axes, e.g., (1,0,0), $\theta < \arccos(1/\sqrt{3})$.
- To exclude points such as (1,1,0), $\theta < \arccos(\sqrt{2}/\sqrt{3})$.

The problem with 'ice-cream' cone...



Consider the following cone given by

$$\mathcal{K}_\theta = \left\{ \mathbf{x} \in \mathbb{R}^N : \frac{\|\mathbf{x}_\parallel^{(\mathbf{1})}\|}{\|\mathbf{x}\|} \geq \cos(\theta) \right\},$$

where $\mathbf{x}_\parallel^{(\mathbf{1})}$ is the projection of $\mathbf{x}$ onto the line $\mathbf{1} = (1, 1, \ldots, 1)$.

Requirements: avoid one queue is empty while others are not.

- To exclude points on axes, e.g., (1,0,0), $\theta < \arccos(1/\sqrt{3})$.
- To exclude points such as (1,1,0), $\theta < \arccos(\sqrt{2}/\sqrt{3})$.
- In general, $\theta < \arccos(\sqrt{N-1}/\sqrt{N})$, which reduces to $\mathbf{1} = (1, 1, \ldots, 1)$ for large N.

Umm...wait, how can we achieve this type of collapse?

Part II: Flexible load balancing

A general view...

The *n*th component of **dispatching distribution** $\mathbf{P}(t)$ is the *probability* of dispatching arrival to the *n*th *shortest* queue.

- let $\sigma_t(\cdot)$ be the permutation of queues in increasing order.
- $P_n(t)$ is then the probability for dispatching to the server $\sigma_t(n)$.

A general view...

The nth component of **dispatching distribution** $\mathbf{P}(t)$ is the *probability* of dispatching arrival to the nth *shortest* queue.

- let $\sigma_t(\cdot)$ be the permutation of queues in increasing order.
- $P_n(t)$ is then the probability for dispatching to the server $\sigma_t(n)$.

We also define dispatching preference

$$\Delta(t) \triangleq \mathbf{P}(t) - \mathbf{P}_{\text{rand}}(t)$$

where $\mathbf{P}_{\text{rand}}(t)$ is the dispatching distribution under random routing.

A general view...

The nth component of **dispatching distribution** $\mathbf{P}(t)$ is the *probability* of dispatching arrival to the nth *shortest* queue.

- let $\sigma_t(\cdot)$ be the permutation of queues in increasing order.
- $P_n(t)$ is then the probability for dispatching to the server $\sigma_t(n)$.

We also define dispatching preference

$$\Delta(t) \triangleq \mathbf{P}(t) - \mathbf{P}_{\text{rand}}(t)$$

where $\mathbf{P}_{\text{rand}}(t)$ is the dispatching distribution under random routing.

- homogeneous servers: the nth component of $\mathbf{P}_{\text{rand}}(t)$ is $1/N$.
- heterogeneous servers: the nth component of $\mathbf{P}_{\text{rand}}(t)$ is $\mu_{\sigma_{t(n)}}/\mu_{\Sigma}$.

Examples...

Consider a system with 4 homogeneous servers.

Examples...

Consider a system with 4 homogeneous servers.

- Random: randomly joins one
  - $\mathbf{P}_{\text{rand}}(t) = (1/4, 1/4, 1/4, 1/4)$
  - $\Delta_{\text{rand}}(t) = (0, 0, 0, 0)$

Examples...

Consider a system with 4 homogeneous servers.

- Random: randomly joins one
  - $\mathbf{P}_{\text{rand}}(t) = (1/4, 1/4, 1/4, 1/4)$
  - $\Delta_{\text{rand}}(t) = (0, 0, 0, 0)$
- JSQ: always join the shortest one
  - $\mathbf{P}_{\text{JSQ}}(t) = (1, 0, 0, 0)$
  - $\Delta_{JSQ}(t) = (3/4, -1/4, -1/4, -1/4)$

Examples...

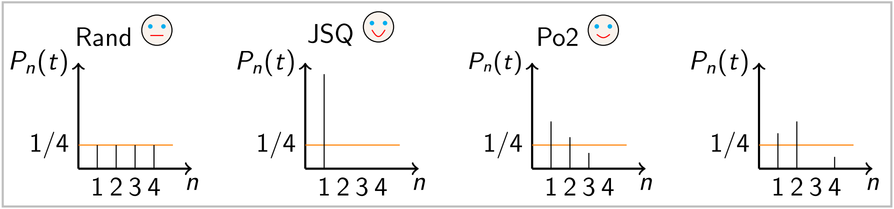
Consider a system with 4 homogeneous servers.

- Random: randomly joins one
  - $\mathbf{P}_{\text{rand}}(t) = (1/4, 1/4, 1/4, 1/4)$
  - $\Delta_{\text{rand}}(t) = (0, 0, 0, 0)$
- JSQ: always join the shortest one
  - $\mathbf{P}_{\text{JSQ}}(t) = (1, 0, 0, 0)$
  - $\Delta_{JSQ}(t) = (3/4, -1/4, -1/4, -1/4)$
- Power of 2: randomly picks two and joins the shorter one
  - $\mathbf{P}_{\text{Po2}}(t) = (1/2, 1/3, 1/6, 0)$
  - $\Delta_{Po2}(t) = (1/4, 1/12, -1/12, -1/4)$

Examples...

Consider a system with 4 homogeneous servers.

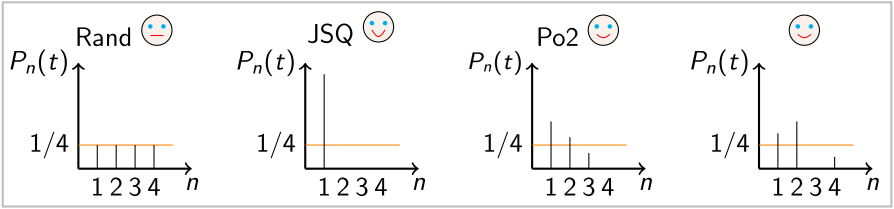
- Random: randomly joins one
  - $\mathbf{P}_{\text{rand}}(t) = (1/4, 1/4, 1/4, 1/4)$
  - $\Delta_{\text{rand}}(t) = (0, 0, 0, 0)$
- JSQ: always join the shortest one
  - $\mathbf{P}_{\text{JSQ}}(t) = (1, 0, 0, 0)$
  - $\Delta_{JSQ}(t) = (3/4, -1/4, -1/4, -1/4)$
- Power of 2: randomly picks two and joins the shorter one
  - $\mathbf{P}_{\text{Po2}}(t) = (1/2, 1/3, 1/6, 0)$
  - $\Delta_{Po2}(t) = (1/4, 1/12, -1/12, -1/4)$



Examples...

Consider a system with 4 homogeneous servers.

- Random: randomly joins one
  - $\mathbf{P}_{\text{rand}}(t) = (1/4, 1/4, 1/4, 1/4)$
  - $\Delta_{\text{rand}}(t) = (0, 0, 0, 0)$
- JSQ: always join the shortest one
  - $\mathbf{P}_{\text{JSQ}}(t) = (1, 0, 0, 0)$
  - $\Delta_{JSQ}(t) = (3/4, -1/4, -1/4, -1/4)$
- Power of 2: randomly picks two and joins the shorter one
  - $\mathbf{P}_{\text{Po2}}(t) = (1/2, 1/3, 1/6, 0)$
  - $\Delta_{Po2}(t) = (1/4, 1/12, -1/12, -1/4)$



Preference of shorter queues...

$$\Delta(t) \triangleq \mathbf{P}(t) - \mathbf{P}_{\mathsf{rand}}(t)$$

Definition

A $\mathbf{P}(t)$ is δ-tilted if, for some $2 \leq k \leq N$

Preference of shorter queues...

$$\Delta(t) \triangleq \mathbf{P}(t) - \mathbf{P}_{\text{rand}}(t)$$

Definition

A $\mathbf{P}(t)$ is δ-tilted if, for some $2 \leq k \leq N$

- $\Delta_n(t) \geq 0$ for all $n < k$ and $\Delta_n(t) \leq 0$ for all $n \geq k$

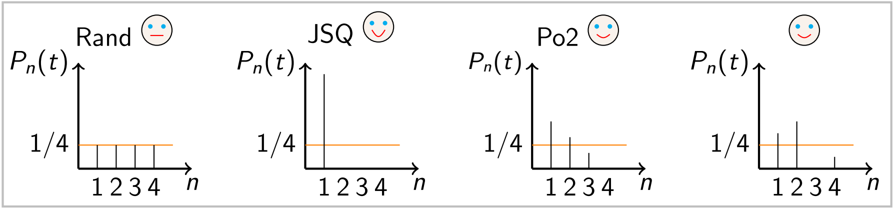
Preference of shorter queues...

$$\Delta(t) \triangleq \mathbf{P}(t) - \mathbf{P}_{\text{rand}}(t)$$

Definition

A $\mathbf{P}(t)$ is δ-tilted if, for some $2 \leq k \leq N$

- $\Delta_n(t) \geq 0$ for all $n < k$ and $\Delta_n(t) \leq 0$ for all $n \geq k$
- $\Delta_1(t) \geq \delta$, $\Delta_N(t) \leq -\delta$



But, where is the cone?

Main Result...

Theorem (δ-tilted outside the cone $\implies$ Optimality)

Main Result...

Theorem (δ-tilted outside the cone $\implies$ Optimality)

Given a load balancing policy,

Main Result...

Theorem (δ-tilted outside the cone $\implies$ Optimality)

Given a load balancing policy, if there exists a cone $\mathcal{K}_\alpha$ *with* $\alpha \in (0, 1]$

Main Result...

Theorem (δ-tilted outside the cone $\implies$ Optimality)

Given a load balancing policy, if there exists a cone $\mathcal{K}_\alpha$ *with* $\alpha \in (0, 1]$ *such that dispatching distribution is* δ*-tilted for any* $\mathbf{Q}(t) \notin \mathcal{K}_\alpha$,

Main Result...

Theorem (δ-tilted outside the cone $\implies$ Optimality)

Given a load balancing policy, if there exists a cone $\mathcal{K}_\alpha$ with $\alpha \in (0,1]$ such that dispatching distribution is δ-tilted for any $\mathbf{Q}(t) \notin \mathcal{K}_\alpha$, then this policy is heavy-traffic delay optimal in steady-state.

Main Result...

Theorem (δ-tilted outside the cone $\implies$ Optimality)

Flexibility from two aspects:

Main Result...

Theorem (δ-tilted outside the cone $\implies$ Optimality)

Flexibility from two aspects:

1. When $\mathbf{Q}(t) \in \mathcal{K}_\alpha$, arbitrary dispatching is allowed.

Main Result...

Theorem (δ-tilted outside the cone $\implies$ Optimality)

Flexibility from two aspects:

1. When $\mathbf{Q}(t) \in \mathcal{K}_\alpha$, arbitrary dispatching is allowed.
2. Preference of shorter queue is not necessarily decreasing.

Main Result...

Theorem (δ-tilted outside the cone $\implies$ Optimality)

Flexibility from two aspects:

1. When $\mathbf{Q}(t) \in \mathcal{K}_\alpha$, arbitrary dispatching is allowed.
2. Preference of shorter queue is not necessarily decreasing.

Applications:

- Load balancing with constraints of *data locality*.
- Load balancing with inaccurate queue lengths information.
- Load balancing with cache replacement cost.
- ......

The challenge to prove it...

Recall that:

Theorem (Stability + Collapse to cone $\implies$ Optimality)

(a) Stability with bounded moments.

The challenge to prove it...

Recall that:

Theorem (Stability + Collapse to cone $\implies$ Optimality)

(a) Stability with bounded moments.

- ☹ standard Foster's theorem is difficult, positive drift in cone.

The challenge to prove it...

Recall that:

Theorem (Stability + Collapse to cone $\implies$ Optimality)

(a) Stability with bounded moments.

- standard Foster's theorem is difficult, positive drift in cone.
- can solve it by combining fluid model with drift analysis.

The challenge to prove it...

Recall that:

Theorem (Stability + Collapse to cone $\implies$ Optimality)

(a) Stability with bounded moments.

- ☹ standard Foster's theorem is difficult, positive drift in cone.
- ☺ can solve it by combining fluid model with drift analysis.

(b) State-space collapses to the cone $\mathcal{K}_\alpha$.

The challenge to prove it...

Recall that:

Theorem (Stability + Collapse to cone $\implies$ Optimality)

(a) Stability with bounded moments.

- ☹ standard Foster's theorem is difficult, positive drift in cone.
- ☺ can solve it by combining fluid model with drift analysis.

(b) State-space collapses to the cone $\mathcal{K}_\alpha$.

- ☹ standard drift-based technique fails in our case.
  - since a closed-form formula of the projection onto a polyhedral cone is still an open problem.

The challenge to prove it...

Recall that:

Theorem (Stability + Collapse to cone $\Longrightarrow$ Optimality)

(a) Stability with bounded moments.

- standard Foster's theorem is difficult, positive drift in cone.
- can solve it by combining fluid model with drift analysis.

(b) State-space collapses to the cone $\mathcal{K}_\alpha$.

- standard drift-based technique fails in our case.
  - since a closed-form formula of the projection onto a polyhedral cone is still an open problem.
- instead, we found that a monotone property of the projection is enough.

Extensions...

Recall that: two parameters determine the flexibility.

- α determines the cone size, and hence how often prefer shorter queues. (frequency)
- δ determines how strong shorter queue is preferred. (intensity)

Extensions...

Recall that: two parameters determine the flexibility.

- α determines the cone size, and hence how often prefer shorter queues. (frequency)
- δ determines how strong shorter queue is preferred. (intensity)

Both of them can scale down to zero with the load to enjoy even greater flexibility.

Extensions...

Recall that: two parameters determine the flexibility.

- α determines the cone size, and hence how often prefer shorter queues. (frequency)
- δ determines how strong shorter queue is preferred. (intensity)

Both of them can scale down to zero with the load to enjoy even greater flexibility.

Proposition

Consider the same policy as before, i.e., δ-tilted outside a cone $\mathcal{K}_\alpha$. Suppose that

$$\alpha^{(\epsilon)}\delta^{(\epsilon)} = \Omega(\epsilon^\beta)$$

for some $\beta \in [0, 1)$, then this policy is heavy-traffic delay optimal.

Geometric intuition...

Proposition

Consider the same policy as before, i.e., δ-tilted outside a cone $\mathcal{K}_\alpha$. Suppose that

$$\alpha^{(\epsilon)}\delta^{(\epsilon)} = \Omega(\epsilon^{\beta})$$

for some $\beta \in [0, 1)$, then this policy is heavy-traffic delay optimal.

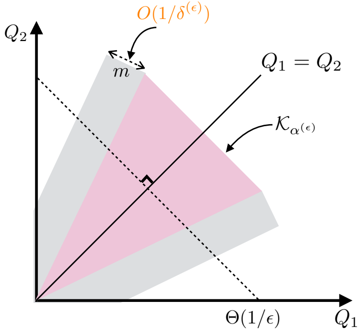
Geometric intuition...

Proposition

Consider the same policy as before, i.e., δ-tilted outside a cone $\mathcal{K}_\alpha$. Suppose that

$$\alpha^{(\epsilon)}\delta^{(\epsilon)} = \Omega(\epsilon^\beta)$$

for some $\beta \in [0, 1)$, then this policy is heavy-traffic delay optimal.



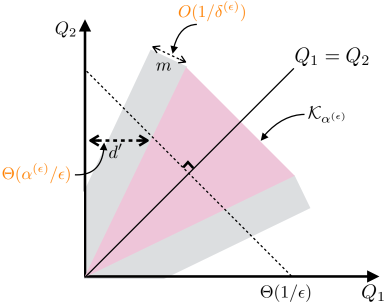
Geometric intuition...

Proposition

Consider the same policy as before, i.e., δ-tilted outside a cone $\mathcal{K}_\alpha$. Suppose that

$$\alpha^{(\epsilon)}\delta^{(\epsilon)} = \Omega(\epsilon^{\beta})$$

for some $\beta \in [0,1)$, then this policy is heavy-traffic delay optimal.



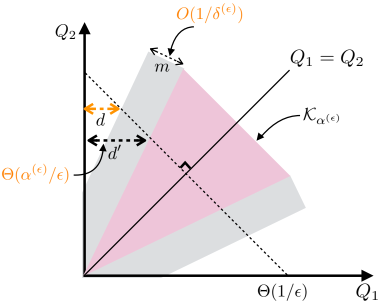
Geometric intuition...

Proposition

Consider the same policy as before, i.e., δ-tilted outside a cone $\mathcal{K}_\alpha$. Suppose that

$$\alpha^{(\epsilon)}\delta^{(\epsilon)} = \Omega(\epsilon^\beta)$$

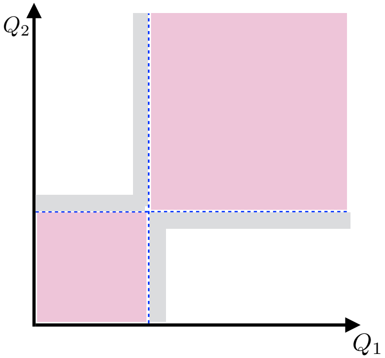
for some $\beta \in [0, 1)$, then this policy is heavy-traffic delay optimal.



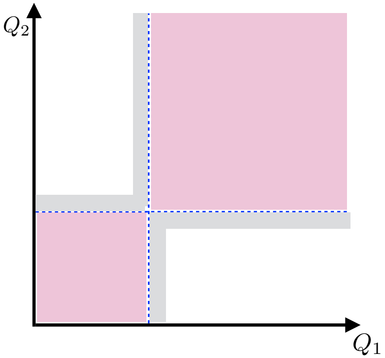
$\lim_{\epsilon\to 0} \frac{d'}{m} = \infty$ far away from boundary

What if... the collapse region cannot be covered by a cone?

What if... the collapse region cannot be covered by a cone?



What if... the collapse region cannot be covered by a cone?



Our new paper addresses it, to appear in Sigmetrics/Performance 2019.

"Heavy-traffic Delay Optimality in Pull-based Load Balancing Systems: Necessary and Sufficient Conditions"

Conclusion...

Theorem (Stability + Collapse to cone $\implies$ Optimality)

- We show a multi-dimensional state-space can still guarantee delay optimality.
- The key is *no sever is idle while others with high loads.*

Theorem (δ-tilted outside the cone $\implies$ Optimality)

- Flexibility comes from two aspects: frequency (α) and intensity (δ).
- The methods to prove the result have the potential in general case.

Thank you!